# Application Layer Protocols for the Internet of Things: A survey

Muneer Bani Yassein
Department of Computer Science .
Jordan University of Science and Technology
Irbid, Jordan
masadeh@just.edu.jo

Mohammed Q. Shatnawi
Department of Computer Inforrmation Systems
Jordan University of Science and Technology
Irbid, Jordan
mshatnawi@just.edu.jo

Dua' Al-zoubi
Department of Computer Science
Jordan University of Science and Technology
Irbid, Jordan
dmalzubi14@cit.just.edu.jo

**Abstract—The "Internet of things" (IoT) concept nowadays is one of the hottest trends for research in any given field; since IoT is about interactions between multiple devices, things, and objects. This interaction opens different directions of enhancement and development in many fields, such as architecture, dependencies, communications, protocols, security, applications and big data. The results will be outstanding and we will be able to reach the desired change and improvements we seek in the fields that affect our lives.**

**The critical goal of Internet of things (IoT) is to ensure effective communication between objects and build a sustained bond among them using different types of applications. The application layer is responsible for providing services and determines a set of protocols for message passing at the application level. This survey addresses a set of application layer protocols that are being used today for IoT, to affirm a reliable tie among objects and things.**

***Keywords— IoT application layer Protocol; XMPP; MQTT; CoAP; RESTFUL; DSS; AMQP; WebSocket***

## I. INTRODUCTION

Many of the devices nowadays adopt the terminology of Internet of Things (IoT) to interconnect; and with this change in interconnection, these devices need different protocols (Bluetooth, Wifi) to avoid the problem of interoperability. The Application Layer – which interacts directly with the end user – consists of applications each with its own application layer protocols, not forget to mention the amount of new protocols that are needed to solve the rising IoT challenges as the old ones don't perform the same.

This survey will introduce the existing application layer protocols in details, which focus basically on message exchange between applications and the internet. This survey also provides a comparison among all discussed protocols based on transport layer used, architecture and communication model. The first section in this proposed paper discusses one of the most common application layer protocols that are used in IoT. The next section shows a comparison between the protocols which are previously discussed in previous sections. Last section suggests some future work to achieve more enhancement in this research area.

## II. CONSTRAINED APPLICATION PROTOCOL (COAP)

Constrained application protocol (CoAP) is request/response protocol; it is similar to client-server model. Nevertheless, this protocol is only sufficient in constrained environment such as: constrained node with low capability in RAM or CPU, and constrained network, such as lower power using wireless personal area network (WPAN). This constrained environment led to bad packet delivery and high overhead. CoAP was designed by Internet Engineering Task Force (IETF) which is mainly interested in machine to machine (m2m) applications and the automation of systems to reduce overhead, enhance packet delivery, and to increase the simplicity of work, by using simple interface with HTTP [1].

CoAP supports publishe/subscribe architecture, this architecture provides multicast communications, and the publisher sends the message so on the other hand multi-subscribers can catch the message and takes the actions. This scenario is done in an Asynchronous way. Publish /subscribe architecture is used to support a large number of users and provide better performance than the traditional way [3].

The most important features in CoAP are simplicity and reliability; since it supports unicast and multicast request by taking advantage of UDP, and provide the ability to Asynchronous message exchanges. CoAP is a single protocol with two layers, the first layer is the messaging layer and the second one is the request/response layer; messaging layer aims to achieve reliability based on UDP, while request/response layer aims to act the interactions and communication.

CoAP uses different types of massages: Conformable Message, Non-conformable Message, Acknowledgement Message, Reset Message, Piggybacked Response, Separate Response, and Empty Message [1] [2]. The following points provide a brief description for each:

978-1-5090-5579-1/16/$31.00

### A. Conformable Message

This type of messages guarantees reliable communication by using the acknowledgment method; if the message arrives at the destination, it should propagate a return message of type acknowledgment or reset message.

### B. Non-conformable Message

In this type there is no need for an acknowledgment message.

### C. Acknowledgment Message

This message means that conformable message arrives.

### D. Reset Message

When a message (conformable, Non-conformable) arrives, but it misses critical and important part required for message interpretation. Propagate resets messages into an empty acknowledgment message.

### E. Piggybacked Response

The receiver responses directly when receiving the message of the acknowledgment message.

### F. Separate Response

The Receiver responses in a different message separate from the acknowledgment message.

Evaluation of CoAP: it works based on Representational State Transfer (REST) architecture, which supports request/response model, such as HTTP. CoAP also supports publish/subscribe architecture, using a Universal Resource Identifier (URI), CoAP applies its own reliability mechanism by using two types of messages; conformable and non-conformable, CoAP has two conceptual sub layers; messaging and request/response layers [4].

CoAP is simple and has lower consumption of CPU and memory. On the other hand though, it is known for its high latency, bad packet delivery, and its inability to be used on complex data type. [5].

## III. Message Queue Telemetry Transport (MQTT)

Message queue telemetry transport (MQTT) is a publishe/subscribe protocol. It's similar to the client-server model. Nevertheless, its simplicity, and open source code make this protocol suited only for constrained environments, such as low power, limited computation capability and memory, and limited bandwidth. It's suitable for IoT applications and machine to machine communications. MQTT protocol can run over TCP/IP. [7].

MQTT was designed by IBM, and by 2013 it was standardized by OASIS [2], it aims to reduce bandwidth requirement. In addition to guarantee reliability of packet delivery, MQTT provides a set of features that includes: the support of multi-cast communication (one to many message), and the capability to establish communications between remote devices. But the most important feature of this protocol is the minimization of network traffic by reducing transport overhead and protocol exchanges. In addition, it provides a notification mechanism when an abnormal situation occurs. [7][8].

MQTT protocol provides three options to achieve messaging Quality of Service (QoS) [8]:

### A. One delivery (at most)

Messages are delivered according to the best effort of the network; an acknowledgment is not required. (Least level of QoS)

### B. One delivery (at least)

Message sends at least once, some duplicate message may exist, and an acknowledgment message is required.

### C. On delivering (exactly)

Require an additional protocol to ensure that the message is delivered once and only once. (Highest level of QoS)

MQTT protocol outperforms CoAP protocol in the case of high traffic network; MQTT provides higher throughput and lower latency than CoAP [9]. The importance of MQTT protocol is due to its simplicity and the no need of high CPU and memory usage (lightweight protocol). MQTT supports a wide range of different devices and mobile platforms. On the other hand, MQTT is high sampling rate and high latency, and dedicated to simple data type only, can't be used in real time applications. [5].

## IV. Extensible Messaging and Presence Protocol (XMPP)

Extensible Messaging and Presence Protocol (XMPP) nowadays is one of the most common communication and messaging protocol in IoT, it was standardized by the IETF. This protocol is a well-known protocol that was used broadly in all networks. The need of IoT can be addressed by XMPP protocol since it supports small messages and low latency; these characteristics make the XMPP protocol a good choice for IoT communications and messaging.

XMPP protocol supports both request/response and publish/subscribe models; request/response which allows bi-directional communications and publishe/subscribe model which allows multi-directional communication (push and pull the data). High scalability in XMPP is provided by decentralized architecture. There are many extensions to XMPP protocol, this allows it to work on the infrastructure-less environment [10].

XMPP protocol uses XML for text communications, this may cause network traffic overhead, but it could be solved by compressing XML using EXI. [11]

XMPP evaluation: it's simple, and can be used in heterogeneous projects and applications. It's an extensible and flexible protocol; many extensions have been defined based on this protocol. But on the other hand, it has some weakness points; since this protocol needs high consumption

of bandwidth and high CPU usage, no guarantee of QoS, and it is restricted to simple data type. [5]

## V. Representational State Transfer (RESTFUL Services)

Representational State Transfer (RESTFUL Services) is an architecture that provides web services which allow communication and data exchange between different devices using HTTP in IoT environment. Simply, this architecture provides resources and access permission from the server, the role of the client is to access and use these resources. Different representations are used in this RESET such as JASON, XML, and text.

Reset architecture uses the same methods used in HTTP, such as GET, PUT, DELETE, POST, and OPTION, to the request or response of resource usage. The methods POST and GET function to create and retrieve resources, while PUT method is to update and change resource state, and DELETE method is to remove the resource. The applications that support RESTFUL web services perform better than others, in addition to that RESTFUL web services are easy to implement and easy to learn [12] [13].

RESTFUL web services support request/response messaging model, by using HTTP commands. RESTFUL web services represent architecture considered as a good choice for IoT since it's supported by the different types of applications. HTTP is a well-known protocol in the World Wide Web, as it provides security since it uses TLS/SSL, along with the reliability of communications in m2m systems [6]. Many testbeds prove the benefits of RESTFUL services for M2M communications in the IoT environment because it provides easy implementation and interaction and it uses an existing and well-known protocol (HTTP) to exchange messages and secure communication [6].

## VI. Advanced Message Queuing Protocol (AMQP)

Advanced Message Queuing Protocol (AMQP) is a publishe/subscribe model which depends on reliable and efficient messaging queue. It's standardized by OASIS. Nowadays, AMQP is widely used in business and commercial platforms. The use of a publish/subscribe approach makes this protocol of high scalability [14].

AMQP supports heterogeneity and interoperability characteristic communications among different devices that support different languages. Applications that belong to AMQP protocol are able to exchange messages one to another. AMQP protocol focuses on knowing a set of the specifications of messages to achieve reliability, security and performance. [15].

AMQP protocol is used in IoT environment which focuses on message exchange, and communication. AMQP uses different message delivery guarantees; at most once, at least once, and exactly once to ensure reliability. This protocol also uses a TCP transport layer to ensure reliability.

Publish/subscribe approach of AMQP consists of two components: exchange queue and message queue, the exchange queue is responsible for message routing to the suitable order in queue. Message queue keeps storing messages until they are sent to the receiver. There is a specific process with a set of rules to exchange messages between exchange components and message queues. [16]

Evaluation of AMQP: this protocol is highly extendable, highly interoperable in different platforms and environments; it has a good ability to support industrial environment applications. On the other hand, it's not suitable for constrained environment and real-time applications, and it does not support an automation discovery mechanism. [11]

## VII. Data Distribution Service (DDS)

Data Distribution Service (DDS) nowadays is an important protocol in IoT environment, and it is a hot topic for research. DDS is working according to publish/subscribe model, it was designed by the Object Management Group (OMG) to support IoT applications and M2M communication. The reason in which of this protocol is a good choice in M2M and IoT refers to its ability to achieve QoS and reliability. This protocol supports different quality of service standards to guarantee reliability. Standards used by this protocol are security, durability, priority... etc. [2].

DDS protocol supports many quality of services (QoS) criteria, depending on publish/subscribe model, which requires an efficient discovery model to help discover the publishers by the subscribers because this is the best measure of the efficiency and reliability of the protocol. [18]. DDS transmits data between publisher and subscriber as which called topic. Data on the topic is generated by the DataReader in subscriber, and DataWritter in publishers. [19].

## VIII. WebSocket

The WebSocket protocol provides two ways for communication between clients and a remote server. WebSocket provides security similar to the security model used in web browsers. This protocol works over TCP and is suitable to the applications that use the browsers and need to interact and communicate with remote hosts. [20].

WebSocket is a web-based protocol that works on the single TCP channel and provides full duplex communications. WebSocket session starts without using the publish/subscribe and request/response approaches as the previous protocols. It depends on building handshake from client to server to start the communication. Once the WebSocket session is established; a full duplex connection in asynchronous manner starts between the client and server, the session keeps running until both client and server end the need for it.

## IX. Comparison and evaluation

This paper discusses the most common application layer protocols, which are used in IoT environment. IoT environment consists of different and wide range scenarios, each scenario supports different environment, different applications, and different devices, and each device varies in

computation capability, also each application needs different requirement to achieve its goal in a desirable manner.

Until now there has been no evaluation includes all the protocols together. Since each protocol may be performed in the best possible manner in a specific scenario, and performs is a bad manner in another. So, choosing a protocol is an application and environment dependent. Since in a constrained communications environment with low power, RESTFUL protocol can perform in an efficient manner, but on the devices that run on battery, MQTT protocol is considered a good choice. When the application needs frequently and a large volume of data updates, publish/ subscribe protocol is then considered a good choice.

Selection of suitable application layer protocol depends on the different factors that are related to devices, application, and the environment. Based on devices, computation and communication ability is an important factor. Regards the environment's low power, constrained communication plays a critical role to select an application layer protocol, in addition to that, it depends on the application itself, different applications mean different needs and requirements, so different applications prefer different protocol.

## X. Conclusion and future work

This paper briefly discusses the most common application layer protocol in IoT environment, and focuses on the evaluation of each protocol in term of the architecture, communication model, security, and achieving the quality of services. It also addresses the weaknesses and strengths for each protocol. This paper provides comprehensive comparison between the existing protocols, so it can help developers and researchers know how to select the suitable protocol for the existing environment and applications.

This survey addresses a brief description of the most recent research about the application layer protocol of IoT environment. In the future work we aim to make an implementation of these protocols and perform sets, simulations and testbeds to a wide range of IoT scenarios that use these protocols to provide an accurate evaluation and comparison. We aspire to provide reliable work to accurately choose and select the most suitable protocol in different environments and applications, so that the work and research be enhanced in the field of IoT and make difference using IoT.

## References


[1] Shelby, Z., Hartke, K., & Bormann, C. (2014). The constrained application protocol ().

[2] Salman, T. Internet of Things Protocols and Standards.

[3] Oh, S., Kim, J. H., & Fox, G. (2010). Real-time performance analysis for publish/subscribe systems. Future Generation Computer Systems, 26 (3), 318-323.

[4] Thangavel, D., Ma, X., Valera, A., Tan, H. X., & Tan, C. K. Y. (2014, April). Performance evaluation of MQTT and CoAP via a common middleware. In Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014 IEEE Ninth International Conference on (pp. 1-6). IEEE.

[5] Talaminos-Barroso, A., Estudillo-Valderrama, M. A., Roa, L. M.rrrr, Reina-Tosina, J., & Ortega-Ruiz, F. (2016). A Machine-to-Machine protocol benchmark for eHealth applications–Use case: Respiratory rehabilitation. Computer Methods and Programs in Biomedicine, 129, 1-11.

[6] Karagiannis, V., Chatzimisios, P., Vazquez-Gallego, F., & Alonso-Zarate, J. (2015). A survey on application layer protocols for the internet of things. Transaction on IoT and Cloud Computing, 3(1), 11-17.

[7] Banks, A., & Gupta, R. (2014). MQTT Version 3.1. 1. OASIS Standard.

[8] Lampkin, V. (2012). What is mqtt and how does it work with websphere mq?

[9] Collina, M., Bartolucci, M., Vanelli-Coralli, A., & Corazza, G. E. (2014, September). Internet of Things application layer protocol analysis over error and delay prone links. In Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC), 2014 7th (pp. 398-404). IEEE.

[10] Bendel, S., Springer, T., Schuster, D., Schill, A., Ackermann, R., & Ameling, M. (2013, March). A service infrastructure for the internet of things based on XMPP. In Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on (pp. 385-388). IEEE.

[11] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. Communications Surveys & Tutorials, IEEE, 17(4), 2347-2376.

[12] Laine, M. (2012). RESTful Web Services for the Internet of Things. Online] Saatavilla: Http://mediaTkk Fi/webservices/personnel/markku_laine/restful_web_services_for _the_internet_of_things Pdf.

[13] Rodriguez, A. (2008). Restful web services: The basics. IBM developerWorks.

[14] Bloebaum, T. H., & Johnsen, F. T. (2015, October). Evaluating publish/subscribe approaches for use in tactical broadband networks. In Military Communications Conference, MILCOM 2015-2015 IEEE (pp. 605-610). IEEE.

[15] Fernandes, J. L., Lopes, I. C., Rodrigues, J. J., & Ullah, S. (2013, July). Performance evaluation of RESTful web services and AMQP protocol. In Ubiquitous and Future Networks (ICUFN), 2013 Fifth International Conference on (pp. 810-815). IEEE.

[16] Standard, O. A. S. I. S. Oasis advanced message queuing protocol (amqp) version 1.0., 2012. URL http://docs. Oasis-open. org/amqp/core/v1. 0/os/amqp-core-complete-v1. 0-os. Pdf.

[17] http://www.omg.org/spec/DDS/1.4/PDF/, accessed in August 2016

[18] An, K., Gokhale, A., Schmidt, D., Tambe, S., Pazandak, P., & Pardo-Castellote, G. (2014, May). Content-based filtering discovery protocol (CFDP): scalable and efficient OMG DDS discovery protocol. In Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems (pp. 130-141). ACM.

[19] Yoon, G., Choi, J., Park, H., & Choi, H. (2016, January). Topic naming service for DDS. In 2016 International Conference on Information Networking (ICOIN) (pp. 378-381). IEEE.

[20] https://tools.ietf.org/html/rfc6455